PORTAL USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: The ACM Digital Library The Guide

object +subtype +supertype +fast SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **object subtype supertype fast** Found **110** of **154,226**

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 110 Result page: **1** 2 3 4 5 6 next

Relevance scale

1 Fast subtype checking in the HotSpot JVM

Cliff Click, John Rose

November 2002 **Proceedings of the 2002 joint ACM-ISCOPE conference on Java Grande**

Full text available: pdf(61.60 KB) Additional Information: full citation, abstract, references, citings, index terms

We present the fast subtype checking implemented in Sun's HotSpot JVM. Subtype checks occur when a program wishes to know if class S implements class T, where S and T are not both known at compile-time. Large Java programs will make millions or even billions of such checks, hence a fast check is essential. In actual benchmark runs our technique performs complete subtype checks in 3 instructions (and only 1 memory reference) essentially all the time. In rare instances it reverts to a slower array ...

Keywords: Java, checkcast, instanceof, subtype, typecase

2 Bidirectional object layout for separate compilation

Andrew C. Myers

October 1995 **ACM SIGPLAN Notices , Proceedings of the tenth annual conference on Object-oriented programming systems, languages, and applications**, Volume 30 Issue 10

Full text available: pdf(1.87 MB) Additional Information: full citation, abstract, references, citings, index terms

Existing schemes for object layout and dispatch in the presence of multiple inheritance and separate compilation waste space and are slower than systems with single inheritance. This paper describes the *bidirectional object layout*, a new scheme for object layout that produces smaller objects and faster method invocations than existing schemes by automatically optimizing particular uses of multiple inheritance. The bidirectional object layout is used for the programming language Theta, and ...

3 Fast algorithms for compressed multimethod dispatch table generation

Eric Dujardin, Eric Amiel, Eric Simon

January 1998 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 20 Issue 1

Full text available: pdf(682.21 KB) Additional Information: full citation, abstract, references, citings, index terms

The efficiency of dynamic dispatch is a major impediment to the adoption of multimethods

in object-oriented languages. In this article, we propose a simple multimethod dispatch scheme based on compressed dispatch tables. This scheme is applicable to any object-oriented language using a method precedence order that satisfies a specific monotonous property (e.g., as Cecil and Dylan) and guarantees that dynamic dispatch is performed in constant time, the latter being a major requirement for so ...

Keywords: dispatch tables, late binding, multimethods, optimization, pole types, run-time dispatch

4 Special issue on persistent object systems: Tigukat: a uniform behavioral objectbase management system

M. Tamer Özsu, Randal Peters, Duane Szafron, Boman Irani, Anna Lipka, Adriana Muñoz

July 1995 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 4 Issue 3

Full text available: pdf(2.78 MB) Additional Information: full citation, abstract, references, citings

We describe the TIGUKAT objectbase management system, which is under development at the Laboratory for Database Systems Research at the University of Alberta. TIGUKAT has a novel object model, whose identifying characteristics include a purely behavioral semantics and a uniform approach to objects. Everything in the system, including types, classes, collections, behaviors, and functions, as well as meta-information, is a first-class object with well-defined behavior. In this way, the model abstr ...

Keywords: database management, objectbase management, persistent storage system, reflective system

5 Fast algorithm for creating space efficient dispatching tables with application to multi-dispatching

Yoav Zibin, Joseph Yossi Gil

November 2002 **ACM SIGPLAN Notices , Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 37 Issue 11

Full text available: pdf(312.23 KB) Additional Information: full citation, abstract, references, citings

The dispatching problem can be solved very efficiently in the single-inheritance~(SI) setting. In this paper we show how to extend one such solution to the multiple-inheritance~(MI) setting. This generalization comes with an increase to the space requirement by a small factor of κ This factor can be thought of as a metric of the complexity of the topology of the inheritance hierarchy.On a data set of~35 hierarchies totaling some~64 thousand types, our dispatching data structure, based on a ...

6 Subtypes vs. where clauses: constraining parametric polymorphism

Mark Day, Robert Gruber, Barbara Liskov, Andrew C. Myers

October 1995 **ACM SIGPLAN Notices , Proceedings of the tenth annual conference on Object-oriented programming systems, languages, and applications**, Volume 30 Issue 10

Full text available: pdf(1.56 MB) Additional Information: full citation, abstract, references, citings, index terms

All object-oriented languages provide support for subtype polymorphism, which allows the writing of generic code that works for families of related types. There is also a need, however, to write code that is generic across types that have no real family relationship. To satisfy this need a programming language must provide a mechanism for parametric polymorphism, allowing for types as parameters to routines and types. We show that to support modular programming and separate compilation there mus ...

7 Efficient type inclusion tests
Jan Vitek, R. Nigel Horspool, Andreas Krall
October 1997 **ACM SIGPLAN Notices , Proceedings of the 12th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 32 Issue 10

Full text available: pdf(2.39 MB) Additional Information: full citation, abstract, references, citings, index terms

A type inclusion test determines whether one type is a subtype of another. Efficient type testing techniques exist for single subtyping, but not for languages with multiple subtyping. To date, the fast constant-time technique relies on a binary matrix encoding of the subtype relation with quadratic space requirements. In this paper, we present three new encodings of the subtype relation, the *packed encoding*, the *bit-packed encoding* and the *compact encoding*. These encodings h ...

8 Network objects
Andrew Birrell, Greg Nelson, Susan Owicki, Edward Wobber
December 1993 **ACM SIGOPS Operating Systems Review , Proceedings of the fourteenth ACM symposium on Operating systems principles**, Volume 27 Issue 5

Full text available: pdf(1.35 MB) Additional Information: full citation, abstract, references, citings, index terms

A network object is an object whose methods can be invoked over a network. This paper describes the design, implementation, and early experience with a network objects system for Modula-3. The system is novel for its overall simplicity. The paper includes a thorough description of realistic marshaling algorithms for network objects.

9 Faster constraint solving with subtypes
Jonathan Edwards, Daniel Jackson, Emina Torlak, Vincent Yeung
July 2004 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2004 ACM SIGSOFT international symposium on Software testing and analysis**, Volume 29 Issue 4

Full text available: pdf(354.36 KB) Additional Information: full citation, abstract, references, index terms

Constraints in predicate or relational logic can be translated into boolean logic and solved with a SAT solver. For faster solving, it is common to exploit the typing of predicates or relations, in order to reduce the number of boolean variables needed to encode the constraint. Here we show how to extend this idea to constraints expressed in a language with subtyping. Our technique, called *atomization*, refactors the type hierarchy into a flat collection of disjoint atomic types. The const ...

Keywords: SAT, analysis, constraints, relational logic, subtypes, verification

10 Keynote address - data abstraction and hierarchy
Barbara Liskov
January 1987 **ACM SIGPLAN Notices , Addendum to the proceedings on Object-oriented programming systems, languages and applications (Addendum)**, Volume 23 Issue 5

Full text available: pdf(1.38 MB) Additional Information: full citation, citings, index terms

11 Efficient subtyping tests with PQ-encoding
Yoav Zibin, Joseph Yossi Gil

October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11

Full text available: pdf(226.81 KB) Additional Information: full citation, abstract, references, citings, index terms

Subtyping test, i.e., determining whether one type is a subtype of another, are a frequent operation during the execution of object-oriented programs. The challenge is in encoding the hierarchy in a small space, while simulataneously making sure that subtyping tests have efficient implmentation. We present a new scheme for encoding multiple and single inheritance hierarchies, which, in the standardized hierarchies, reduces the footprint of all previsously published schemes. The scheme is ...

12 Converting java programs to use generic libraries

Alan Donovan, Adam Kiežun, Matthew S. Tschantz, Michael D. Ernst

October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN Conference on Object-oriented programming, systems, languages, and applications**, Volume 39 Issue 10

Full text available: pdf(1.18 MB) Additional Information: full citation, abstract, references, index terms

Java 1.5 will include a type system (called JSR-14) that supports parametric polymorphism, or generic classes. This will bring many benefits to Java programmers, not least because current Java practice makes heavy use of logically-generic classes, including container classes.

Translation of Java source code into semantically equivalent JSR-14 source code requires two steps: parameterization (adding type parameters to class definitions) and instantiation (a ...

Keywords: JSR-14, Java 1.5, Java 5, generic types, instantiation types, parameterized types, parametric polymorphism, raw types, type inference

13 Typechecking and modules for multi-methods

Craig Chambers, Gary T. Leavens

October 1994 **ACM SIGPLAN Notices , Proceedings of the ninth annual conference on Object-oriented programming systems, language, and applications**, Volume 29 Issue 10

Full text available: pdf(2.28 MB) Additional Information: full citation, abstract, references, citings, index terms

Two major obstacles hindering the wider acceptance of multi-methods are concerns over the lack of encapsulation and modularity and the absence of static typechecking in existing multi-method-based languages. This paper addresses both of these problems. We present a polynomial-time static typechecking algorithm that checks the conformance, completeness, and consistency of a group of method implementations with respect to declared message signatures. This algorithm improves on previous algori ...

14 Converting Java classes to use generics

Daniel von Dincklage, Amer Diwan

October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN Conference on Object-oriented programming, systems, languages, and applications**, Volume 39 Issue 10

Full text available: pdf(259.67 KB) Additional Information: full citation, abstract, references, index terms

Generics offer significant software engineering benefits since they provide code reuse without compromising type safety. Thus generics will be added to the Java language in the next release. While this extension to Java will help programmers when they are writing new

code, it will not help legacy code unless it is rewritten to use generics. In our experience, manually modifying existing programs to use generics is complex and can be error prone and labor intensive.

We describe a system ...

Keywords: generics, parametric polymorphism, type inference

15 Escape analysis for Java™: Theory and practice

Bruno Blanchet

November 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 6

Full text available: pdf(684.21 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Escape analysis is a static analysis that determines whether the lifetime of data may exceed its static scope.This paper first presents the design and correctness proof of an escape analysis for Java™. This analysis is interprocedural, context sensitive, and as flow-sensitive as the static single assignment form. So, assignments to object fields are analyzed in a flow-insensitive manner. Since Java is an imperative language, the effect of assignments must be precisely determined. Thi ...

Keywords: Java, optimization, stack allocation, static analysis, synchronization elimination

16 Typechecking and modules for multimethods

Craig Chambers, Gary T. Leavens

November 1995 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 17 Issue 6

Full text available: pdf(2.90 MB) Additional Information: full citation, references, citings, index terms

Keywords: encapsulation, inheritance, multimethods, static typechecking, subtyping

17 Ownership types for object encapsulation

Chandrasekhar Boyapati, Barbara Liskov, Liuba Shrira

January 2003 **ACM SIGPLAN Notices , Proceedings of the 30th ACM SIGPLAN-SIGACT symposium on Principles of programming languages**, Volume 38 Issue 1

Full text available: pdf(307.67 KB) Additional Information: full citation, abstract, references, citings, index terms

Ownership types provide a statically enforceable way of specifying object encapsulation and enable local reasoning about program correctness in object-oriented languages. However, a type system that enforces strict object encapsulation is too constraining: it does not allow efficient implementation of important constructs like iterators. This paper argues that the right way to solve the problem is to allow objects of classes defined in the same module to have privileged access to each other's re ...

Keywords: object encapsulation, ownership types, software upgrades

18 Static analysis to support the evolution of exception structure in object-oriented systems

Martin P. Robillard, Gail C. Murphy

April 2003 **ACM Transactions on Software Engineering and Methodology (TOSEM)**, Volume 12 Issue 2

Full text available: pdf(708.48 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Exception-handling mechanisms in modern programming languages provide a means to help software developers build robust applications by separating the normal control flow of a program from the control flow of the program under exceptional situations. Separating the exceptional structure from the code associated with normal operations bears some consequences. One consequence is that developers wishing to improve the robustness of a program must figure out which exceptions, if any, can flow to a po ...

Keywords: Error handling, exception flow, exception handling, exception structure, program evolution, static analysis

19 Type elaboration and subtype completion for Java bytecode

Todd B. Knoblock, Jakob Rehof

March 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 23 Issue 2

Full text available: pdf(873.69 KB) Additional Information: full citation, abstract, references, citings, index terms

Java source code is strongly typed, but the translation from Java source to bytecode omits much of the type information originally contained within methods. Type elaboration is a technique for reconstructing strongly typed programs from incompletely typed bytecode by inferring types for local variables. There are situations where, technically, there are not enough types in the original type hierarchy to type a bytecode program. Subtype completion is a technique for adding necessary types to ...

Keywords: Java compiler, lattice completion, object-oriented type systems, type inference, type reconstruction, type-directed compilation, typed intermediate language

20 On objects and events

Patrick Th. Eugster, Rachid Guerraoui, Christian Heide Damm

October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11

Full text available: pdf(308.58 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents linguistic primitives for publish/subscribe programming using events and objects. We integrate our primitives into a strongly typed object-oriented language through four mechnisms: (1) serialization, (2) multiple subtyping, (3)closures, and (4) deferred code evaluation. We illustrate our primitives through Java, showing how we have overcome its respective lacks. A precompiler transforms statements based on our publish/subscribe primitives into calls to specifically generated ...

Results 1 - 20 of 110 Result page: **1** 2 3 4 5 6 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

Google Web Images Groups News Froogle Local more »
object subtyping supertype check Search Advanced Search Preferences

Web Results **1** - **10** of about **6,720** for **object subtyping supertype check**. (**0.27** seconds)

Visio-Based Database Modeling in Visual Studio .NET Enterprise ...
... Basic **Subtyping**. If the population of an **object** type A must always be a ...
and a list box for selecting the primary **supertype**. The **check** box titled ...
msdn.microsoft.com/.../html/ vstchVisio-BasedDatabaseModelingInVisualStudioNETEnterpriseArchitectPart4.asp
- 26k - Cached - Similar pages

[PDF] 6.170 Lecture 10 Subtypes
File Format: PDF/Adobe Acrobat - View as HTML
... should be consistent with an **object** of the **supertype**. being manipulated ...
List<**Object**> is not. Also have **subtyping** relationship with raw types ...
www.mit.edu/~6.170/lectures/lect10-**subtyping**.pdf - Similar pages

Cecil **Object** Model
... conform to the subtype are guaranteed also to conform to the **supertype**. ...
specify the inheritance and **subtyping** properties of an **object**/type pair for ...
www.objs.com/x3h7/cecil.htm - 22k - Cached - Similar pages

SENG 609.03 **Object** Theory: Research Paper
... 3. Sharing in **Object**-Oriented Systems. 3.1 Inheritance 3.2 **Subtyping** ...
There is the possibility to **check** at compile time if a special **object** knows a ...
sern.ucalgary.ca/courses/ SENG/609.03/W98/hongd2/essay.html - 41k - Cached - Similar pages

JOT: Journal of **Object** Technology - The Theory of Classification
... simple **object** types and has so far developed a theory of **subtyping** which judges
... A pointer can be coerced to a **supertype** without difficulty and the ...
www.jot.fm/issues/issue_2003_03/column2 - 40k - Cached - Similar pages

Generalization
... Gets the **supertype** of this generalization. Returns:: The **supertype object**.
check. public boolean **check**(). Checks if well-formedness rules are respected ...
elmuth.tigris.org/nonav/doc/javadocs/ UML/Semantic/Foundation/Core/Generalization.html - 21k -
Cached - Similar pages

Modula-3: **Object** types
... It is more often used for the opposite purpose, when it requires a runtime **check**.
The last example uses **object subtyping** to define reusable queues. ...
research.compaq.com/SRC/m3defn/html/**objects**.html - 10k - May 10, 2005 - Cached - Similar pages

[PPT] **Subtyping** and Inheritance
File Format: Microsoft Powerpoint 97 - View as HTML
... **Subtyping**. Inheritance. What is **Object**-Oriented Programming ... To implement
a subtype, it is often useful to use the implementation of its **supertype** ...
www.cs.virginia.edu/cs201j/lectures/lecture12.ppt - Similar pages

[PPT] Metamodel for **Object** Role Modeling
File Format: Microsoft Powerpoint 97 - View as HTML
... **Subtyping** relationships are depicted as solid arrows from subtype to **supertype**.
An **object** type either is primitive or is a subtype ...
moncs.cs.mcgill.ca/people/hv/teaching/ MSBDesign/presentations/050314.BillSong.ppt - Similar pages

Google Web Images Groups News Froogle Local more »
object subtyping overflow Search Advanced Search Preferences

Web Results **1 - 10** of about **799** for **object subtyping overflow**. (**0.14** seconds)

merd vs other languages
... In merd, the default is to use bignums and bigrats which can't **overflow** (nor
... The records do not allow **subtyping**. The **object's** value is a record, ...
merd.sourceforge.net/merd-vs-others.html - 16k - Cached - Similar pages

[PPT] Semantic Consistency in Information Exchange
File Format: Microsoft Powerpoint 97 - View as HTML
... Space **overflow** is not really the right condition to trigger finalization when
... An exception is an **object** from an exception class. **Subtyping** between ...
www.stanford.edu/class/cs242/slides/2003/java.ppt - Similar pages

[PDF] 1 Java Outline History Gates Saw Java as Real Threat Design Goals ...
File Format: PDF/Adobe Acrobat - View as HTML
... Space **overflow** is not really the right condition to trigger ... An exception
is an **object** from an exception class. • **Subtyping** between exception classes ...
www.stanford.edu/class/cs242/slides/2003/java.pdf - Similar pages

[PDF] Principles and a Preliminary Design for ML2000
File Format: PDF/Adobe Acrobat - View as HTML
... **Object** types follow a structural **subtyping** discipline. Both width **subtyping** (longer
object types are subtypes of. shorter **object** types) and depth ...
flint.cs.yale.edu/flint/publications/ml2000.pdf - Similar pages

JOT: JOT: Journal of **Object** Technology - Theory of Classification ...
... What is the difference between **subtyping** and subclassing? ... codes caused
arithmetic **overflow** in the data conversion intended for the propulsion system ...
www.jot.fm/issues/issue_2002_05/column5 - 32k - Cached - Similar pages

[PPT] Bytecode Verification on Java Smart Cards
File Format: Microsoft Powerpoint 97 - View as HTML
... Treats all interface types as class type "**Object**". Verifier contains proper
classes and **subtyping** between them is simply inheritance relation ...
www.cis.ksu.edu/~tamtoft/890Sec/Presentations/Leroy.ppt - Similar pages

EP1324194
... (n) means for searching, wherein an **overflow** array is searched for an **object**
subtyping data structure corresponding to the input **object** type; ...
swpat.ffii.org/pikta/txt/ep/1324/194/ - 50k - Cached - Similar pages

[PDF] PLUG AND PLAY SAFELY: RULES FOR BEHAVIOURAL COMPATIBILITY
File Format: PDF/Adobe Acrobat - View as HTML
... **Object** extension with **subtyping** is explained in the same. formal framework,
supporting the ... were to **overflow** and lead to the Error state, there would ...
www.dcs.shef.ac.uk/~ajhs/papers/plugplay.pdf - Similar pages

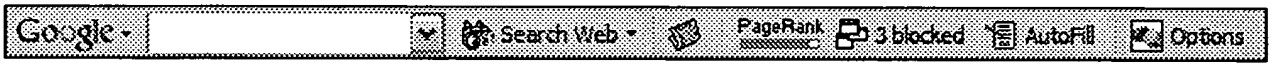
[PDF] An Approach to Behavioral **Subtyping** Based on Static Analysis
File Format: PDF/Adobe Acrobat - View as HTML
... if the B's **object** did not. Behavioral **subtyping** tries to overcome this problem.
[7,10]. ... eg division by zero, **overflow** or null-pointer dereferencing, ...

www.enseignement.polytechnique.fr/ profs/informatique/Francesco.Logozzo/tacos04.pdf - Similar pages

[PDF] Simple objects for Standard ML
File Format: PDF/Adobe Acrobat
... **object**. **subtyping**. Our mechanism. is based on a new declar-. ative construcc. objtype ... mental conflict between **subtyping** (in which an **object** may ...
portal.acm.org/ft_gateway.cfm?id=231412&type=pdf - Similar pages

Gooooooooogle ▶
Result Page: 1 2 3 4 5 6 7 8 9 10 Next

Free! Get the Google Toolbar. Download Now - About Toolbar



object subtyping overflow Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google

©2005 Google

IEEE Xplore RELEASE 2.0

Search Result - Print Format

< Back t

Key: IEEE JNL = IEEE Journal or Magazine, IEE JNL = IEE Journal or Magazine, IEEE CNF = IEEE Conference, I
CNF = IEE Conference, IEEE STD = IEEE Standard

1. **Implementing functional languages on object-oriented virtual machines**
 Perry, N.; Meijer, E.;
 Software, IEE Proceedings- [see also Software Engineering, IEE Proceedings]
 Volume 151, Issue 1, 9 Feb. 2004 Page(s):1 - 9
 IEE JNL

2. **Design and implementation of assertions for the Common Language Infrastructure**
 Tran, N.; Mingins, C.; Abramson, D.;
 Software, IEE Proceedings- [see also Software Engineering, IEE Proceedings]
 Volume 150, Issue 5, 27 Oct. 2003 Page(s):329 - 336
 IEE JNL

3. **A straightforward approach to effort estimation for updating programs in object-oriented prototyping development**
 Uehara, S.; Mizuno, O.; Kikuno, T.;
 Software Engineering Conference, 1999. (APSEC '99) Proceedings. Sixth Asia Pacific
 7-10 Dec. 1999 Page(s):144 - 151
 IEEE CNF

4. **Mixin' up components**
 Sreedhar, V.C.;
 Software Engineering, 2002. ICSE 2002. Proceedings of the 24rd International Conference on
 2002 Page(s):198 - 207
 IEEE CNF

5. **Understanding any IDL lesson one: DCE and CORBA**
 Vogel, A.; Gray, B.; Duddy, K.;
 Services in Distributed and Networked Environments, 1996., Proceedings of Third International Workshop on
 3-4 June 1996 Page(s):114 - 121
 IEEE CNF

6. **Fuzzy types and their lattices**
 Cao, T.H.; Creasy, P.N.; Wuwongse, V.;
 Fuzzy Systems, 1997., Proceedings of the Sixth IEEE International Conference on
 Volume 2, 1-5 July 1997 Page(s):805 - 812 vol.2
 IEEE CNF

Indexed by Inspec

© Copyright 2005 IEEE –

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L2	1	("20030126305").PN.	US-PGPUB; USPAT	OR	OFF	2005/05/12 10:51
L3	1	2 and (computer adj readable adj medium)	US-PGPUB; USPAT	OR	ON	2005/05/12 10:54
L4	1	2 and ((computer adj readable) with (signal or carrier))	US-PGPUB; USPAT	OR	ON	2005/05/12 10:56
L5	1	2 and ((signal or carrier))	US-PGPUB; USPAT	OR	ON	2005/05/12 10:56
L6	19	object with type with (sub adj typing or subtyp$) same (supertyp$ or super adj typing)	US-PGPUB; USPAT	OR	ON	2005/05/12 12:33
L7	68	(object and typ$3 and cach$).ti, ab.	US-PGPUB; USPAT; USOCR	OR	ON	2005/05/12 12:36
L8	1	(object and typing and cach$).ti, ab.	US-PGPUB; USPAT; USOCR	OR	ON	2005/05/12 12:36
L9	56	(object and (typing or subtyp$ or supertyp$)).ti,ab.	US-PGPUB; USPAT; USOCR	OR	ON	2005/05/12 13:03
L10	2142	719/315-316.ccls. or 717/114,116.ccls. or 707/103$.ccls.	USPAT	OR	ON	2005/05/12 13:08
L11	1	object with type same supertype same embedd$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:09
L12	3	object with type same supertype same cach$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:09
L13	119	object with type with (check$ or determin$ or verify$) same cach$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:09
L14	45	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$)	US-PGPUB; USPAT	OR	ON	2005/05/12 13:29
L15	13	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (search$3 or look$3 or find$3 or check$3 or test$3 or locat$3)	US-PGPUB; USPAT	OR	ON	2005/05/12 13:17
L16	1	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (hierarchy or tree) same cach$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:12
L17	2	object same typ$3 same (sub-typ$ or subtyp$) same (hierarchy or tree) same cach$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:13
L18	96	object same typ$3 same (hierarchy or tree) same cach$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:13

L19	6	10 and 18	US-PGPUB; USPAT	OR	ON	2005/05/12 13:13
L20	11	9 and 10	US-PGPUB; USPAT	OR	ON	2005/05/12 13:21
L21	9	7 and 10	US-PGPUB; USPAT	OR	ON	2005/05/12 13:22
L22	468	type adj hierarchy	US-PGPUB; USPAT	OR	ON	2005/05/12 13:27
L23	123	L22 and cach$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:28
L24	3	L22 and cach$ and overflow$	US-PGPUB; USPAT	OR	ON	2005/05/12 13:28
L25	330471	object with (typ$3 or (sub-typ$ or subtyp$) or (supertyp$ or super-typ$))	US-PGPUB; USPAT	OR	ON	2005/05/12 13:29
L26	2266	object with (typ$3 or (sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) with (cach$ or overflow$ or embedd$)	US-PGPUB; USPAT	OR	ON	2005/05/12 13:34
L27	47	(object with (typ$3 or (sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) with (cach$ or overflow$ or embedd$)).ti,ab.	US-PGPUB; USPAT	OR	ON	2005/05/12 13:33
L28	108	(object with (typ$3 or (sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) with (cach$ or overflow$ or embedd$)).clm.	US-PGPUB; USPAT	OR	ON	2005/05/12 13:31
L29	7	(object with (typ$3 or (sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) with (cach$ or overflow$ or embedd$) with (data adj structure)).clm.	US-PGPUB; USPAT	OR	ON	2005/05/12 13:32
L30	1	(object with (typ$3 or (sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) with (cach$ or overflow$ or embedd$) with (array)).clm.	US-PGPUB; USPAT	OR	ON	2005/05/12 13:32
L31	2	(object with (typ$3 or (sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) with (cach$ or overflow$ or embedd$) and array).ti,ab.	US-PGPUB; USPAT	OR	ON	2005/05/12 13:33
L32	5	object with ((sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) with (cach$ or overflow$ or embedd$)	US-PGPUB; USPAT	OR	ON	2005/05/12 13:35
L33	24	object same ((sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) same (cach$ or overflow$ or embedd$)	US-PGPUB; USPAT	OR	ON	2005/05/12 13:38

L34	10	object same ((sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) same (cach$ or overflow$ or embedd$)	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 13:45
L35	0	(object same ((sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) same (cach$ or overflow$ or embedd$)).clm.	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 14:42
L36	51	Rose$.in. and (object and typ$3).ti,ab.	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 14:43
L37	2	Rose-J$.in. and (object and typ$3).ti,ab.	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 14:44
L38	164	(Sun adj Microsystem$).as. and (object and typ$3).ti,ab.	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 14:47
L39	5	(Sun adj Microsystem$).as. and (object and typ$3 and subtyp$3).ti,ab.	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/05/12 14:47
L40	6	(Sun adj Microsystem$).as. and (object and typ$3 and subtyp$3).ti,ab.	US-PGPUB; USPAT	OR	ON	2005/05/12 14:47
L41	109	(Sun adj Microsystem$).as. and (object and typ$3).ti,ab.	US-PGPUB; USPAT	OR	ON	2005/05/12 14:47
L42	6	(Sun adj Microsystem$).as. and (object and typ$3 and subtyp$3).ti,ab.	US-PGPUB; USPAT	OR	ON	2005/05/12 14:48
L43	7	(Sun adj Microsystem$).as. and (object and typ$3 and (subtyp$3 or supertype)).ti,ab.	US-PGPUB; USPAT	OR	ON	2005/05/12 14:48

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S7	2	S4 and S6	US-PGPUB; USPAT; USOCR	OR	ON	2004/12/17 11:30
S8	1	(object and typing and cach$).ti, ab.	US-PGPUB; USPAT; USOCR	OR	ON	2005/05/12 12:35
S9	4	object with typing same cach$	US-PGPUB; USPAT; USOCR	OR	ON	2004/12/17 11:34
S10	1696	object with type same cach$	US-PGPUB; USPAT; USOCR	OR	ON	2004/12/17 11:34
S11	51	(object and (typing or subtyp$ or supertyp$)).ti,ab.	US-PGPUB; USPAT; USOCR	OR	ON	2004/12/17 11:35
S12	7	S10 and S11	US-PGPUB; USPAT; USOCR	OR	ON	2004/12/17 11:35
S13	5	("5267349" \| "5437027" \| "5579518" \| "5793963" \| "5915253").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2004/12/17 11:41
S14	3	("5267349").URPN.	USPAT	OR	ON	2004/12/17 12:02
S15	1125	719/315-316.ccls. or 717/114,116.ccls.	USPAT	OR	ON	2004/12/17 12:03
S16	2039	719/315-316.ccls. or 717/114,116.ccls. or 707/103$.ccls.	USPAT	OR	ON	2005/03/16 10:30
S17	34	S3 and S16	USPAT	OR	ON	2004/12/17 12:03
S18	9	S3 and S16 and cach$	USPAT	OR	ON	2004/12/17 12:10
S19	1	("6711516").PN.	US-PGPUB; USPAT	OR	OFF	2004/12/17 12:11
S20	1	("6711517").PN.	US-PGPUB; USPAT	OR	OFF	2004/12/17 12:11
S21	1	("6711576").PN.	US-PGPUB; USPAT	OR	OFF	2004/12/17 14:00
S22	1	object with type same supertype same embedd$	US-PGPUB; USPAT	OR	ON	2004/12/17 14:00
S23	2	object with type same supertype same cach$	US-PGPUB; USPAT	OR	ON	2004/12/17 14:07
S24	262	object with type same(check$ or determin$ or verify$) same cach$	US-PGPUB; USPAT	OR	ON	2004/12/17 14:08
S25	104	object with type with (check$ or determin$ or verify$) same cach$	US-PGPUB; USPAT	OR	ON	2004/12/17 14:08
S26	12	S16 and S25	US-PGPUB; USPAT	OR	ON	2004/12/17 14:08

S27	1	("5377323").PN.	US-PGPUB; USPAT	OR	OFF	2004/12/19 11:49
S28	24	object with type with (sub-typ$ or subtyp$) same (supertyp$ or super-typ$)	US-PGPUB; USPAT	OR	ON	2005/03/15 15:07
S29	9	(("6687760") or ("5793965") or ("6631478") or ("20020021788") or ("5819283") or ("6427123") or ("4945475") or ("20030014442") or ("20030065874")).PN.	US-PGPUB; USPAT	OR	OFF	2005/03/15 14:18
S30	8	("5327562" \| "5404525" \| "5632034" \| "5721824" \| "5913064" \| "5918052" \| "6102966" \| "6526571").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2005/03/15 14:47
S31	234	class adj cache	US-PGPUB; USPAT; USOCR	OR	ON	2005/03/15 14:48
S32	40	class adj cache same object	US-PGPUB; USPAT; USOCR	OR	ON	2005/03/15 14:52
S33	42	subtype same supertype same object	US-PGPUB; USPAT; USOCR	OR	ON	2005/03/15 14:53
S34	4	subtype same supertype same object same array	US-PGPUB; USPAT; USOCR	OR	ON	2005/03/15 14:58
S35	6	subtype same supertype same array	US-PGPUB; USPAT; USOCR	OR	ON	2005/03/15 14:58
S36	42	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$)	US-PGPUB; USPAT	OR	ON	2005/03/15 15:57
S37	11	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (search$3 or look$3 or find$3 or check$3 or test$3)	US-PGPUB; USPAT	OR	ON	2005/03/15 15:18
S38	13	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (search$3 or look$3 or find$3 or check$3 or test$3 or locat$3)	US-PGPUB; USPAT	OR	ON	2005/03/15 15:35
S39	5	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (search$3 or look$3 or find$3 or check$3 or test$3 or locat$3) same (embedd$3 or cach$3)	US-PGPUB; USPAT	OR	ON	2005/03/15 15:37

S40	17	object same typ$3 same (sub-typ$ or subtyp$) same (embedd$3 or cach$3)	US-PGPUB; USPAT	OR	ON	2005/03/15 15:37
S41	5	("5267349" \| "5437027" \| "5579518" \| "5793963" \| "5915253").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2005/03/15 15:45
S42	1	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same overflow	US-PGPUB; USPAT	OR	ON	2005/03/15 15:58
S43	0	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same constant	US-PGPUB; USPAT	OR	ON	2005/03/15 15:58
S44	4	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same known	US-PGPUB; USPAT	OR	ON	2005/03/15 16:36
S45	1	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (hierarchy or tree) same cache	US-PGPUB; USPAT	OR	ON	2005/05/12 13:12
S46	1	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (hierarchy or tree) same cach$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:37
S47	1	object same typ$3 same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (hierarchy or tree) same embedd$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:41
S48	101	object same typ$3 same (hierarchy or tree) same embedd$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:37
S49	24	object with typ$3 with (hierarchy or tree) same embedd$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:39
S50	8	object with typ$3 with (hierarchy or tree) with embedd$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:38
S51	12	object with typ$3 with (hierarchy) same embedd$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:40
S52	6	object with typ$3 with (hierarchy) with cach$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:40
S53	1	object same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (hierarchy or tree) same embedd$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:41
S54	1	object same (sub-typ$ or subtyp$) same (supertyp$ or super-typ$) same (hierarchy or tree) same cach$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:42

S55	3	object same ((sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) same (hierarchy or tree) same cach$3	US-PGPUB; USPAT	OR	ON	2005/03/15 16:44
S56	0	object same ((sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) same (hierarchy or tree) same cach$3	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/03/15 16:43
S57	0	object same typing same (hierarchy or tree) same cach$3	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/03/15 16:43
S58	6	object same type same (hierarchy or tree) same cach$3	EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/03/15 16:43
S59	1	object same ((sub-typ$ or subtyp$) or (supertyp$ or super-typ$)) same (hierarchy or tree) same overflow	US-PGPUB; USPAT	OR	ON	2005/03/15 16:44
S60	6	(("5761493") or ("6389425") or ("6032151") or ("6182083") or ("4420744") or ("5793965")).PN.	US-PGPUB; USPAT	OR	OFF	2005/03/16 09:49
S61	767	phantom.ti,ab.	US-PGPUB; USPAT	OR	ON	2005/03/16 09:50
S62	5	object with type with hierarchy with cache	US-PGPUB; USPAT	OR	ON	2005/03/16 09:52
S63	1	object with type with hierarchy with overflow	US-PGPUB; USPAT	OR	ON	2005/03/16 09:53
S64	4	object with type with hierarchy with partition$3	US-PGPUB; USPAT	OR	ON	2005/03/16 09:56
S65	0	object with type adj hierarchy with partition$3	US-PGPUB; USPAT	OR	ON	2005/03/16 09:56
S66	2	object with type adj hierarchy with cach$	US-PGPUB; USPAT	OR	ON	2005/03/16 10:30
S67	438	type adj hierarchy	US-PGPUB; USPAT	OR	ON	2005/03/16 10:31
S68	19177	719/315-316.ccls. or "717"/$.ccls. or "707"/$.ccls.	USPAT	OR	ON	2005/03/16 10:31
S69	103	S67 and S68	USPAT	OR	ON	2005/03/16 10:31
S70	13	type adj hierarchy.ti,ab.	US-PGPUB; USPAT	OR	ON	2005/03/16 10:39
S71	1	S70 and cach$	US-PGPUB; USPAT	OR	ON	2005/03/16 10:42
S72	2	S70 and (embedd$ or in adj memory or type adj library)	US-PGPUB; USPAT	OR	ON	2005/03/16 10:43
S73	0	S70 and (overflow)	US-PGPUB; USPAT	OR	ON	2005/03/16 10:42